IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Patent Application for:

METHOD AND APPARATUS FOR CONTROLLING SET-TOP BOX HARDWARE AND SOFTWARE FUNCTIONS

Inventor(s): Toshiro Ozawa

Docket Number: SNY-P4255.01

Prepared By:

Miller Patent Services
29 Seminole Drive
Ringwood, NJ 07456

Phone: (973) 728-2760
Fax: (973) 728-0438
Email: miller@patent-inventions.com

CERTIFICATE OF EXPRESS MAILING FOR NEW PATENT APPLICATION

"Express Mail" mailing label number EK726740269US

Date of Deposit 2/14/2001

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231.

JERRY A. MILLER
(Typed or printed name of person mailing paper or fee)

(Signature of person mailing paper or fee)

METHOD AND APPARATUS FOR CONTROLLING SET-TOP BOX HARDWARE AND SOFTWARE FUNCTIONS

CROSS REFERENCE TO RELATED DOCUMENTS

This application is related to and claims priority benefit of U. S. provisional patent application serial number 60/197,233, filed April 14, 2000 to Pedlow, Jr. et al. entitled "Cable Modem Set-Top Box", U. S. provisional patent application serial number 60/197,848, filed April 14, 2000 to Scanlan entitled "A User Interface for a Set-Top Box", and U. S. provisional patent application serial number 60/237,570, filed October 3, 2000 to Toshiro Ozawa entitled "Browser-to-Middleware Interface Using HTTP Microserver", attorney docket number 50P4255, which are hereby incorporated herein by reference.

COPYRIGHT NOTICE

FIELD OF THE INVENTION

This invention relates generally to the field of television set-top boxes. More particularly, this invention relates to television set-top boxes using a browser interface and to a method and apparatus of controlling various hardware and software set-top box functions through the browser.

BACKGROUND OF THE INVENTION

Television set-top boxes are used to provide tuning capabilities for cable and satellite television systems. While these devices still provide that fundamental function, digital set-top boxes now often incorporate powerful computers in the latest generation of set-top boxes. With such computers available, it is now possible to expand the usefulness of the television set-top box beyond that of merely providing tuning functions for cable and satellite systems. Many set-top boxes now provide users with the ability to navigate the Internet using their television sets rather than a computer.

An on-screen user interface (UI) is commonly used with televisions and television set-top boxes for cable and satellite television systems to support numerous features. Features such as menus of favorite channels, locks and limits, timed recording, etc. as well as electronic program guides, pay-per-view (PPV) or video-on-demand (VOD) and other menus and guides are typically provided through such a user interface. In most instances, the user interface screens provided by televisions and television set-top boxes are bit map graphics with all of the behavior (navigation, state transition, etc.) implemented as software programs. Such programs are often coded in the "C" programming language or a variation thereof.

As the functionality of the user interface increases, the user interface structure becomes more complicated and the amount of coding required to support the on-screen user interface increases. Moreover, the amount of memory required by the graphics images grows and may become a cost issue due to the need to provide more memory within the device. In addition, even minor changes to the user interface specification generally require a code change making the design of such user interfaces extremely difficult to modify and inflexible.

In order to ameliorate these problems, some set-top boxes are being designed with Internet access capability. HTML (Hypertext Markup Language) user interface pages and a web browser are used to provide the user interface. This approach provides the advantages of reduced memory size to store user interface screens since HTML files are small compared with bit map graphics, flexibility to adopt changes since changing the user interface page design only requires changing the HTML files, and ease of creating a menu structure by using dynamic links. However, these advantages can be overshadowed by the disadvantages that the browser itself has a large memory footprint. Also, since the browser is optimized for Internet access, accessing local functions are not generally accommodated.

HTML by its very nature is designed to protect the resources of client computer from the outside. It is generally not possible to create an HTML file so that the loading of the file will directly access software outside the browser. In the case of an electronic programming guide, for example, it is easy to display a program guide page written in HTML but difficult to tune a channel by selecting a button on the program guide that requires access to local functions inside the set-top box.

In order to accommodate such functions requiring access to local resources within the set-top box, the interface between the browser and the remaining set-top box can be customized within the browser. This can be accomplished, for example, by defining special URLs that are trapped by the browser to call functions to lower level software (such as middleware or device drivers) before they are processed as normal URLs by sending an HTTP request to a server, for example. The special URL can be described as follows:

```
internal:function?<parameter1>value</parameter1><parameter2>value2</parameter2>
```

In this example the key word "internal" is trapped by the browser and the remainder of the string is translated to an internal function with two parameters. However, this solution also has drawbacks. Due to the requirement of modifying the browser source code, which is often produced by another party, in-depth knowledge of the code and access to the code is required. In addition, the special URL code required to implement this modification is outside the HTML standard. Finally, one URL is needed for each function to be accessed. For each URL addition the entire browser software generally requires recompiling.

An HTTP Microserver is sometimes provided as a part of a real time operating system. Such HTTP Microservers function in a manner similar to any other HTTP Server and are conventionally used to provide for remote system setup and diagnostics to permit access to internal API calls from outside computers via a TCP/IP network. Sometimes HTTP Microservers are also used to provide a display mechanism for a device that normally has no display of its own. Thus, a personal computer with a browser can be used to access a microserver on the device to monitor status or change settings. This function is used on devices such as printers and routers. In summary, HTTP Microservers conventionally provide the ability to access functionality from the outside of a device.

SUMMARY OF THE INVENTION

The present invention relates generally to set-top boxes. Objects, advantages and features of the invention will become apparent to those skilled in the art upon consideration of the following detailed description of the invention.

In one embodiment of the present invention a method and apparatus for controlling set-top box hardware and software functions uses an HTTP microserver to intercept HTTP requests for access to local devices and software. When the HTTP microserver receives a URLfrom the TCP/IP network stack, it directs instructions to an appropriate interface module that in turn generates an API call for an appropriate middleware module. The middleware module may then appropriately control a lower level software module to effect a software function or to drive hardware.

In accordance with an embodiment consistent with the present invention, a method of controlling local hardware or software using a browser includes: directing an HTTP request from a browser to a local HTTP microserver having an IP address; at the HTTP microserver, parsing the HTTP request to identify a target interface module, and directing the HTTP request to the target interface module; and at the target interface module, generating an API call from the HTTP request.

Another method, consistent with the present invention, of controlling local hardware or software using a browser includes: directing a request from a browser to a local microserver having an address; at the microserver, parsing the request to identify a target interface module, and directing the request to the target interface module; and at the target interface module, generating an application call from the request.

In another embodiment, a television set-top box consistent with an embodiment of the invention includes a programmed processor. A browser software segment runs on the programmed processor. A user interface software segment also runs on the programmed processor and receives a user command to select a link using the browser software segment. A network stack receives messages directed to an IP address from the browser software segment in response to the user command selecting a link, and issues an HTTP request in response thereto directed to the IP address. A middleware software module also runs on the programmed processor. An HTTP microserver has an IP address and runs as a software segment on the programmed processor. The HTTP microserver has an interface module that interfaces with the middleware software module by issuing an API (Application Program Interface) call to the middleware software module in response to the HTTP request, the API call implementing the user command.

In another exemplary embodiment, a television set-top box includes a programmed processor. A browser software segment runs on the programmed processor. A user interface software segment also runs on the programmed processor and receives user commands to select a link using the browser software segment. A network stack receives messages directed to an IP address from the browser software segment in response to user commands that select selecting links, and issues HTTP requests in response thereto directed to the IP address. A plurality of middleware software modules also run on the programmed processor. An HTTP microserver having an IP address also runs as a software segment on the programmed processor. The HTTP microserver has a plurality of interface modules that interface with the plurality of middleware software modules by issuing API calls to the plurality of middleware software module in response to the HTTP request, the API calls implementing the user commands.

Another embodiment of a television set-top box consistent with the invention includes a programmed processor. A browser software segment runs on the programmed processor. A user interface software segment also runs on the programmed processor and receives user commands to select a link using the browser software segment. A network stack receives messages directed to an address from the browser software segment in response to user commands that select selecting links, and issues requests in response thereto directed to the address. A plurality of middleware software modules also run on the programmed processor. A microserver having the address also runs as a software segment on the programmed processor. The microserver has a plurality of interface modules that interfaces with the plurality of middleware software modules by issuing application calls to the plurality of middleware software module in response to the request, the application calls implementing the user commands.

Another television set-top box consistent with the invention includes a programmed processor. A browser software segment runs on the programmed processor. A user interface software segment runs on the programmed processor and receives a user command to select a link using the browser software segment. A TCP/IP network stack receives messages directed to an IP address from the browser software segment in response to the user command selecting a link, and issues an HTTP request in response thereto directed to the IP address. A middleware software module also runs on the programmed processor. An HTTP microserver having an IP address also runs as a software segment on the programmed processor. The HTTP microserver has an interface module that interfaces with the middleware software module by issuing an API call to the middleware software module in response to the HTTP request, the API call implementing the user command. A television tuner hardware driver interfaces to and controls the television tuner hardware driver. A television tuner is provided wherein the user command includes a command to change a selected television channel. The API call directs the middleware software module to change channels and the middleware software module directs the television tuner driver to change a channel tuned by the television tuner.

In yet another embodiment, a television set-top box includes a programmed processor. A browser software segment runs on the programmed processor. A user interface software segment also runs on the programmed processor and receives a user command to select a link using the browser software segment. A TCP/IP network stack receives messages directed to an IP address from the browser software segment in response to the user command selecting a link and issues an HTTP request in response thereto directed to the IP address. A middleware software module also runs on the programmed processor. An HTTP microserver having an IP address also runs as a software segment on the programmed processor. The HTTP microserver has an interface module that interfaces with the middleware module by issuing an API call to the middleware software module in response to the HTTP request, the API call implementing the user command. A segment of lower level software code segment carries out one of a memory write and a memory read operation under the direction of the API call.

In another embodiment, an electronic storage medium storing instructions which, when executed on a programmed processor, carry out a process of controlling local hardware or software using a browser including: directing an HTTP request from a browser to a local HTTP microserver having an IP address; at the HTTP microserver, parsing the HTTP request to identify a target interface module, and directing the HTTP request to the target interface module; and at the target interface module, generating an API call from the HTTP request.

Certain embodiments of the present invention provide a method and apparatus for controlling set-top box hardware and software functions. An HTTP microserver is used to intercept HTTP requests for access to local devices and software. When the HTTP microserver receives a URLfrom the TCP/IP network stack, it directs instructions to an appropriate interface module that in turn generates an API call for an appropriate middleware module. The middleware module may then appropriately control a lower level software module to effect a software function or to drive hardware.

The above summaries are intended to illustrate exemplary embodiments of the invention, which will be best understood in conjunction with the detailed description to follow, and are not intended to limit the scope of the appended claims.

BRIEF DESCRIPTION OF THE DRAWINGS

The features of the invention believed to be novel are set forth with particularity in the appended claims. The invention itself however, both as to organization and method of operation, together with objects and advantages thereof, may be best understood by reference to the following detailed description of the invention, which describes certain exemplary embodiments of the invention, taken in conjunction with the accompanying drawings in which:

FIGURE 1 is a block diagram of the relevant software of an exemplary set-top box using an embodiment of the present invention.

FIGURE 2 is a flow chart of an embodiment of the software process carried out in the browser and TCP/IP stack of an embodiment of the present invention.

FIGURE 3 is a flow chart of an embodiment of the software process carried out in an HTTP microserver, consistent with embodiments of the invention, and beyond as a result of an HTTP request received by the HTTP microserver.

DETAILED DESCRIPTION OF THE INVENTION

While this invention is susceptible of embodiment in many different forms, there is shown in the drawings and will herein be described in detail specific embodiments, with the understanding that the present disclosure is to be considered as an example of the principles of the invention and not intended to limit the invention to the specific embodiments shown and described. In the description below, like reference numerals are used to describe the same, similar or corresponding parts in the several views of the drawings.

For purposes of this application, the term "lower level software" is intended to embrace any software code segment that operates under the direction of a middleware software layer. The term "local" is intended to embrace hardware and software devices that are a part of the set-top box or are coupled directly to the set-top box and are thus directly addressable by the set-top box. The term "local" is intended to exclude hardware and software that is addressable via an IP address over the Internet.

Referring now to **FIGURE 1**, a set-top box 100 utilizes an HTTP (Hypertext Transfer Protocol) microserver 104 to implement the present invention. HTTP microservers are often provided as part of the real time operating system package. Such microservers are primarily used for remote systems set up in diagnostics to enable access to internal API (Application Program Interface) calls from outside computers via the TCP/IP network. It has basically the same functionality that a regular HTTP server provides. In response to HTTP requests, the HTTP microserver 104 executes server functions and returns the results in an HTML (Hypertext Markup Language) file. The HTTP microserver 104 can form a part of the set-top box software running as a separate task (or equivalently, running on another processor operating in concert). It is possible to access the HTTP microserver 104 from a browser 106 forming a part of the same set-top box software. Such access is accomplished via the set-top boxes TCP/IP network stack 110. This makes it possible for the browser 106 to indirectly access the set-top box 100 software via the HTTP microserver 104. By use of such HTTP microserver, there is no need to modify the browser 106 in order to accomplish control of set-top box hardware and software functions. Although such modifications might require modification to the HTTP microserver, the HTTP microserver 104 code is substantially smaller than that of the browsers code and such code is designed to be easily interfaced with lower level software. Thus, it is substantially easier to modify the microserver then to modify the browser 106.

In operation, a user provides input to the set-top box by interacting with a user interface 114 formed by displaying an HTML page using browser 106. Such HTML page may be contained within an HTML file 118 accessed by the browser 106. When the user interacts with the browser 106 to select a particular link to a Universal Resource Locator (URL), the URLs are transferred to the TCP/IP network stack 110. If the URL address is a normal Internet resource, an HTTP request is transmitted over the Internet 120 in order to access the desired URL. If, however, the URL such as URL1 and URL 2 are directed to a local address, the HTTP request is directed to HTTP microserver 104. Such HTTP requests can take the form of:

http://localhost/dir/func?parameter1=value1&parameter2=value2

wherein “localhost” is the HTTP microservers IP address, “dir” is a file location and “parameter1" and “parameter2" are provided with “value1" and “value2" to be operated upon by the HTTP microserver 104. In one example, parameter1 and parameter2 could represent a television channel number and a name for the channel with the function being a function defining a favorite channel function. In this manner, the HTTP microserver 104 can make a call to lower level software to write to a memory location establishing a channel number and channel name for a favorite channel desired by the user. In other embodiments, parameter1 could represent a channel that the user desires to select with the function being a channel changing function. In this case, the HTTP microserver 104 would interface with a middleware module and hardware driver designed to control television or set-top box tuner. It will be clear to those skilled in the art that numerous other set-top box and television and video appliance functions can be implemented using similar processes.

When the HTTP microserver 104 receives an HTTP request from the TCT/IP network stack 110, an HTTP request parser 126 forming a part of the HTTP microserver directs the HTTP request to a selected one of a plurality of interface modules such as interface module 130 or interface module 131 illustrated. In the case of interface module 130, this interface module is designed to interface with a middleware module 136 via an application program interface call. Middleware module 136 then controls lower level software 140 to implement any number of functions such as memory reads and memory writes that might affect audio or video characteristics for example. In another example, the HTTP request parser 126 may direct an HTTP request to interface module 131 that is designed to interface with middleware module 150 and control a hardware driver 156. This hardware driver is simply another form of lower level software that can implement control over hardware such as hardware device 160. By way of example, and not limitation, driver 156 may be a driver that controls a tuner in the set-top box for implementing channel selection. If the HTTP request is for a channel change, interface module 131 sends an appropriate API call to middleware module 150 which, in turn, causes driver 156 to select a selected channel using tuner hardware 160. Those skilled in the art will appreciate that other hardware can also be controlled using the present invention. Examples of such hardware might include a cable modem, a dial-up modem, a serial or parallel interface or any other suitable hardware forming a part of the set-top box or connected to the set-top box.

Referring now to **FIGURE 2**, a process 200 is described starting at 210 with process 200 describing the actions leading up to a HTTP request being sent to HTTP microserver 104. The processor waits the user selection of a particular link at 216. Once that link is selected by the user, a URL is sent to the TCP/IP network stack at 222. If the address in the URL is a local IP address at 228 such as that representing the HTTP microserver, then the TCP/IP network stack 110 sends an HTTP request to microserver 104 at 236 and control returns to 216 until the user selects another link. In the case where the address at 228 is not a local IP address, the network stack sends the HTTP request to the Internet 120 in a normal fashion.

Once an HTTP request is sent to HTTP microserver 104 at 236, process 300 of **FIGURE 3** is carried out starting at 310. When a request is received from the network stack at 316, the request is parsed by the HTTP request parser 126 and directed to an appropriate interface module at 324. The interface module receiving the request then makes an API call to the appropriate middleware module or alternatively directly to a driver at 332. Depending upon whether the middleware module is ultimately to control hardware or software at 340, a command is either passed to lower level software at 346 or to a hardware driver at 352. If the command is passed to the hardware driver at 352, the driver controls the hardware at 358. In either event, control returns to 316 to await the next request from the TCP/IP network stack at 316.

Of course many variations are possible without departing from the invention. The multiple interface modules 130 through 131 described in conjunction with **FIGURE 1** may be implemented as a single interface module or may be implemented as many interface modules as depicted. Moreover, similar implementation can be carried out for the middleware modules 136 and 150. Other variations will occur to those skilled in the art including variations of use of other mark-up languages and other protocols besides HTML and TCP/IP as described herein. While the invention has been described in conjunction with an interface to a television set-top box, similar functionality can be implemented within a television set itself. Such an implementation should be considered equivalent.

Those skilled in the art will recognize that the present invention has been described in terms of exemplary embodiments based upon use of a programmed processor. However, the invention should not be so limited, since the present invention could be implemented using hardware component equivalents such as special purpose hardware and/or dedicated processors which are equivalents to the invention as described and claimed. Similarly, multiple processors operating in concert, general purpose computers, microprocessor based computers, micro-controllers, optical computers, analog computers, dedicated processors and/or dedicated hard wired logic may be used to construct alternative equivalent embodiments of the present invention.

Those skilled in the art will appreciate that the program steps used to implement the embodiments described above can be implemented using disc storage as well as other forms of storage including Read Only Memory (ROM) devices, Random Access Memory (RAM) devices; optical storage elements, magnetic storage elements, magneto-optical storage elements, flash memory, core memory and/or other equivalent storage technologies without departing from the present invention. Such alternative storage devices should be considered equivalents.

The present invention is preferably implemented using a programmed processor executing programming instructions that are broadly described above in flow chart form which can be stored on any suitable electronic storage medium. However, those skilled in the art will appreciate that the processes described above can be implemented in any number of variations and in many suitable programming languages without departing from the present invention. For example, the order of certain operations carried out can often be varied, and additional operations can be added without departing from the invention. Error trapping can be added and/or enhanced and variations can be made in user interface and information presentation without departing from the present invention. Such variations are contemplated and considered equivalent.

While the invention has been described in conjunction with specific embodiments, it is evident that many alternatives, modifications, permutations and variations will become apparent to those skilled in the art in light of the foregoing description. Accordingly, it is intended that the present invention embrace all such alternatives, modifications and variations as fall within the scope of the appended claims.

What is claimed is: